2007

Extraction of text regions in natural images

Sneha Sharma

Extraction of Text Regions in Natural Images

Sneha Sharma
Dr. Roxanne Canosa, advisor

Masters Project Report

Department of Computer Science

Rochester Institute of Technology

Spring 2006/07

Approved By:

Project Advisor
Dr. Roxanne Canosa

Reader
Dr. Rajendra Raj

Observer
Dr. Hans Peter Bischof

Abstract

The detection and extraction of text regions in an image is a well known problem in the computer vision research area. The goal of this project is to compare two basic approaches to text extraction in natural (non-document) images: edge-based and connected-component based. The algorithms are implemented and evaluated using a set of images of natural scenes that vary along the dimensions of lighting, scale and orientation. Accuracy, precision and recall rates for each approach are analyzed to determine the success and limitations of each approach. Recommendations for improvements are given based on the results.

Table of Contents

1. Introduction

Recent studies in the field of computer vision and pattern recognition show a great amount of interest in content retrieval from images and videos. This content can be in the form of objects, color, texture, shape as well as the relationships between them. The semantic information provided by an image can be useful for content based image retrieval, as well as for indexing and classification purposes [4,10]. As stated by Jung, Kim and Jain in [4], text data is particularly interesting, because text can be used to easily and clearly describe the contents of an image. Since the text data can be embedded in an image or video in different font styles, sizes, orientations, colors, and against a complex background, the problem of extracting the candidate text region becomes a challenging one [4]. Also, current Optical Character Recognition (OCR) techniques can only handle text against a plain monochrome background and cannot extract text from a complex or textured background [7].

Different approaches for the extraction of text regions from images have been proposed based on basic properties of text. As stated in [7], text has some common distinctive characteristics in terms of frequency and orientation information, and also spatial cohesion. *Spatial cohesion* refers to the fact that text characters of the same string appear close to each other and are of similar height, orientation and spacing [7]. Two of the main methods commonly used to determine spatial cohesion are based on edge [1,2] and connected component [3] features of text characters.

The fact that an image can be divided into categories depending on whether or not it contains any text data can also be used to classify candidate text regions. Thus other methods for text region detection, as described in more detail in the following section, utilize classification techniques such as support vector machines [9,11], k-means clustering [7] and neural network based classifiers [10]. The algorithm proposed in [8] uses the focus of attention mechanism from visual perception to detect text regions.

2. Related Work

The purpose of this project is to implement, compare, and contrast the edge-based and the connected component methods. The other methods mentioned here are examples of text extraction techniques that can be used for future projects.

Various methods have been proposed in the past for detection and localization of text in images and videos. These approaches take into consideration different properties related to text in an image such as color, intensity, connected-components, edges etc. These properties are used to distinguish text regions from their background and/or other regions within the image. The algorithm proposed by Wang and Kangas in [5] is based on color clustering. The input image is first pre-processed to remove any noise if present. Then the image is grouped into different color layers and a gray component. This approach utilizes the fact that usually the color data in text characters is different from the color data in the background. The potential text regions are localized using connected component based heuristics from these layers. Also an aligning and merging analysis (AMA) method is

used in which each row and column value is analyzed [5]. The experiments conducted show that the algorithm is robust in locating mostly Chinese and English characters in images; some false alarms occurred due to uneven lighting or reflection conditions in the test images.

The text detection algorithm in [6] is also based on color continuity. In addition it also uses multi-resolution wavelet transforms and combines low as well as high level image features for text region extraction. The textfinder algorithm proposed in [7] is based on the frequency, orientation and spacing of text within an image. Texture based segmentation is used to distinguish text from its background. Further a bottom-up 'chip generation' process is carried out which uses the spatial cohesion property of text characters. The chips are collections of pixels in the image consisting of potential text strokes and edges. The results show that the algorithm is robust in most cases, except for very small text characters that are not properly detected. Also in the case of low contrast in the image, misclassifications occur in the texture segmentation.

A focus of attention based system for text region localization has been proposed by Liu and Samarabandu in [8]. The intensity profiles and spatial variance is used to detect text regions in images. A Gaussian pyramid is created with the original image at different resolutions or scales. The text regions are detected in the highest resolution image and then in each successive lower resolution image in the pyramid.

The approach used in [9, 11] utilizes a support vector machine (SVM) classifier to segment text from non-text in an image or video frame. Initially text is detected in multi

scale images using edge based techniques, morphological operations and projection profiles of the image [11]. These detected text regions are then verified using wavelet features and SVM. The algorithm is robust with respect to variance in color and size of font as well as language.

3. Approach

The goal of the project is to implement, test, and compare and contrast two approaches for text region extraction in natural images, and to discover how the algorithms perform under variations of lighting, orientation, and scale transformations of the text. The algorithms are from Liu and Samarabandu in [1,2] and Gllavata, Ewerth and Freisleben in [3]. The comparison is based on the accuracy of the results obtained, and precision and recall rates. The technique used in [1,2] is an edge-based text extraction approach, and the technique used in [3] is a connected-component based approach.

In order to test the robustness and performance of the approaches used, each algorithm was first implemented in the original proposed format. The algorithms were tested on the image data set provided by Xiaoqing Liu (xliu65@uwo.ca) and Jagath Samarabandu (jagath@uwo.ca), as well as another data set which consists of a combination of indoor and outdoor images taken from a digital camera. The results obtained were recorded based on criteria such as invariance with respect to lighting conditions, color, rotation, and distance from the camera (scale) as well as horizontal and/or vertical alignment of text in an image. The experiments have also been conducted for images containing

different font styles and text characters belonging to language types other than English. Also, the precision and recall rates (Equations (1) and (2)), have been computed based on the number of correctly detected words in an image in order to further evaluate the efficiency and robustness of each algorithm.

The Precision rate is defined as the ratio of correctly detected words to the sum of correctly detected words plus false positives. *False positives* are those regions in the image which are actually not characters of a text, but have been detected by the algorithm as text regions.

$$\text{Precision Rate} = \frac{\text{Correctly detected words}}{\text{Correctly detected words} + \text{False positives}} \times 100\% \qquad (1)$$

The Recall rate is defined as the ratio of correctly detected words to the sum of correctly detected words plus false negatives. *False Negatives* are those regions in the image which are actually text characters, but have not been detected by the algorithm.

$$\text{Recall Rate} = \frac{\text{Correctly detected words}}{\text{Correctly detected words} + \text{False Negatives}} \times 100\% \qquad (2)$$

3.1 Algorithm for edge based text region extraction [1,2]

The basic steps of the edge-based text extraction algorithm are given below, and diagrammed in Figure 1. The details are explained in the following sections.

1. Create a Gaussian pyramid by convolving the input image with a Gaussian kernel and successively down-sample each direction by half. (Levels: 4)
2. Create directional kernels to detect edges at 0, 45, 90 and 135 orientations.
3. Convolve each image in the Gaussian pyramid with each orientation filter.
4. Combine the results of step 3 to create the Feature Map.
5. Dilate the resultant image using a sufficiently large structuring element (7x7 [1]) to cluster candidate text regions together.
6. Create final output image with text in white pixels against a plain black background.

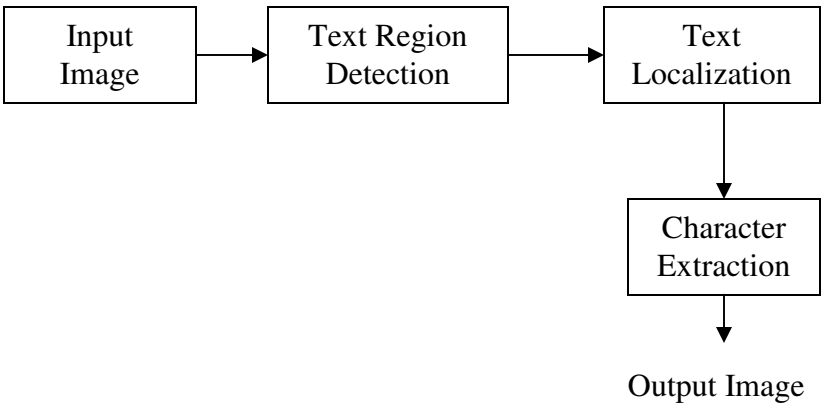
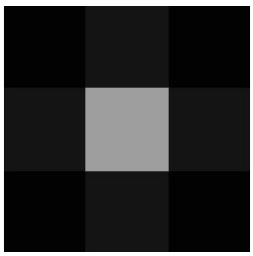


Figure 1. Basic Block diagram for edge based text extraction.

As given in [1][2], the procedure for extracting a text region from an image can be broadly classified into three basic steps: (1)detection of the text region in the image, (2)localization of the region, and (3) creating the extracted output character image.

3.1.1 Detection

This section corresponds to Steps 1 to 4 of 3.1. Given an input image, the region with a possibility of text in the image is detected [1,2]. A Gaussian pyramid is created by successively filtering the input image with a Gaussian kernel of size 3x3 and down-sampling the image in each direction by half. Down sampling refers to the process whereby an image is resized to a lower resolution from its original resolution. A Gaussian filter of size 3x3 will be used as shown in Figure 2. Each level in the pyramid corresponds to the input image at a different resolution. A sample Gaussian pyramid with 4 levels of resolution is shown in Figure 3. These images are next convolved with directional filters at different orientation kernels for edge detection in the horizontal (0°), vertical (90°) and diagonal (45°, 135°) directions. The kernels used are shown in Figure5.



0.0113	0.0838	0.0113
0.0838	0.6193	0.0838
0.0113	0.0838	0.0113

Figure 2. Default filter returned by the fspecial Gaussian function in Matlab. Size [3 3], Sigma 0.5

Figure 3. Sample Gaussian pyramid with 4 levels

Figure 4. Each resolution image resized to original image size

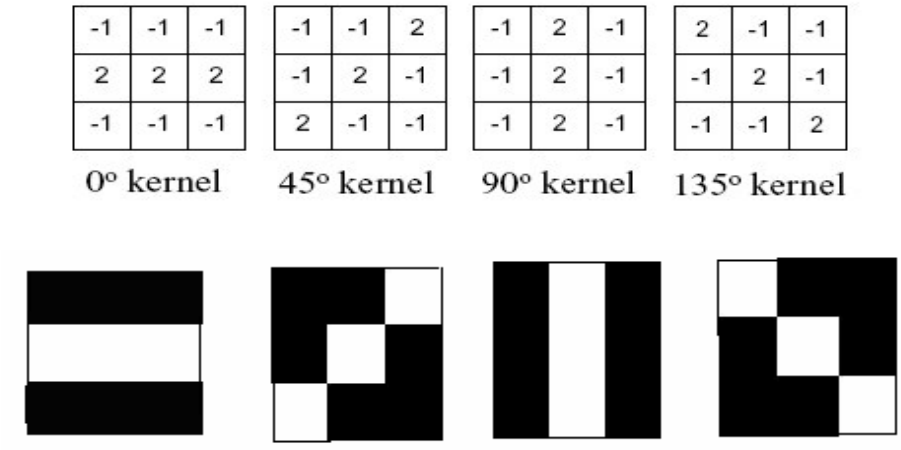
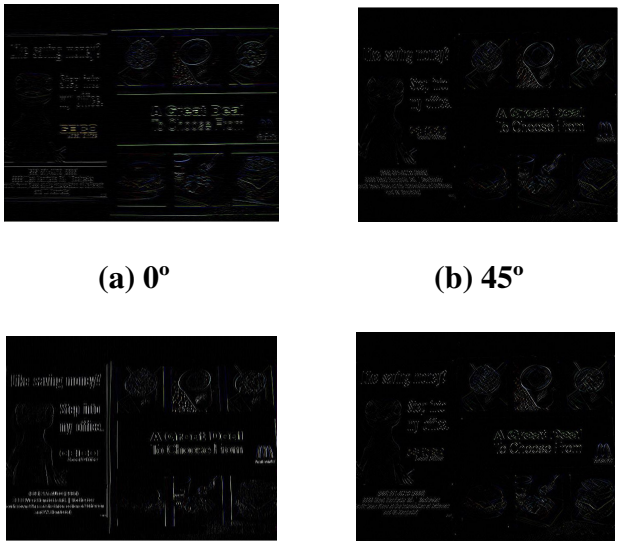


Figure 5. The directional kernels [1]



(a) 0º **(b) 45º**

(c) 90º **(d) 135º**

Figure 6. Sample image from Figure 3 after convolution with each directional kernel

Note how the edge information in each direction is highlighted.

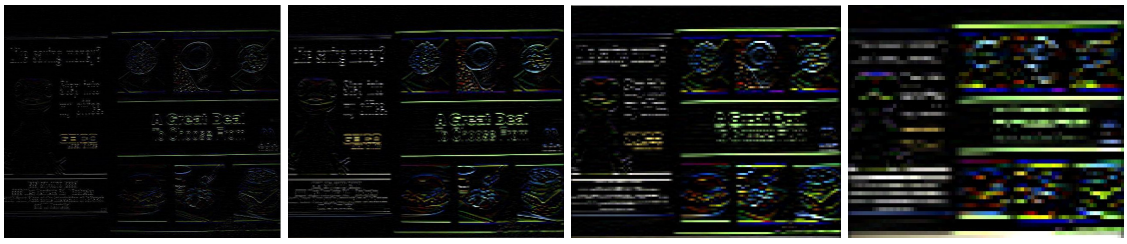


Figure 7. Sample resized image of the pyramid after convolution with 0º kernel

After convolving the image with the orientation kernels, a feature map is created. A weighting factor is associated with each pixel to classify it as a candidate or non-candidate for text region. A pixel is a candidate for text if it is highlighted in all of the edge maps created by the directional filters. Thus, the feature map is a combination of all

edge maps at different scales and orientations with the highest weighted pixels present in the resultant map.

3.1.2. Localization

This section corresponds to Step 5 of 3.1. The process of localization involves further enhancing the text regions by eliminating non-text regions [1,2]. One of the properties of text is that usually all characters appear close to each other in the image, thus forming a cluster. By using a morphological dilation operation, these possible text pixels can be clustered together, eliminating pixels that are far from the candidate text regions. *Dilation* is an operation which expands or enhances the region of interest, using a structural element of the required shape and/or size. The process of dilation is carried out using a very large structuring element in order to enhance the regions which lie close to each other. In this algorithm, a structuring element of size [7x7] has been used [1]. Figure 8 below shows the result before and after dilation.

(a) **(b)**

Figure 8. (a) Before dilation (b) After dilation

The resultant image after dilation may consist of some non-text regions or noise which needs to be eliminated. An area based filtering is carried out to eliminate noise blobs present in the image. According to [1], only those regions in the final image are retained which have an area greater than or equal to 1/20 of the maximum area region.

3.1.3 Character extraction

This section corresponds to Step 6 of 3.1. The common OCR systems available require the input image to be such that the characters can be easily parsed and recognized. The text and background should be monochrome and background-to-text contrast should be high [3]. Thus this process generates an output image with white text against a black background [1,2]. A sample test image [1,2] and its resultant output image from the edge based text detection algorithm are shown in Figures 9(a) and 9(b) below.

(a) **(b)**

Figure 9. (a) Original image [1,2] (b) Result

3.2 Algorithm for Connected Component based text region extraction [3]

The basic steps of the connected-component text extraction algorithm are given below, and diagrammed in Figure 10. The details are discussed in the following sections.

1. Convert the input image to YUV color space. The luminance(Y) value is used for further processing. The output is a gray image.
2. Convert the gray image to an edge image.
3. Compute the horizontal and vertical projection profiles of candidate text regions using a histogram with an appropriate threshold value.
4. Use geometric properties of text such as width to height ratio of characters to eliminate possible non-text regions.
5. Binarize the edge image enhancing only the text regions against a plain black background.
6. Create the Gap Image (as explained in the next section) using the gap-filling process and use this as a reference to further eliminate non-text regions from the output.

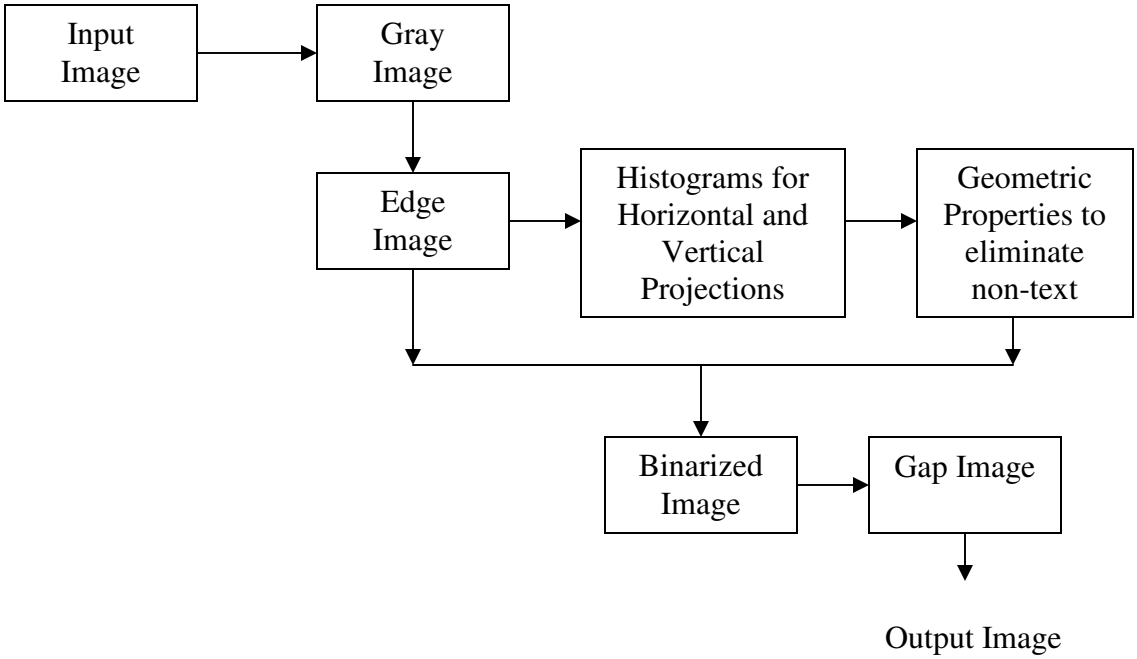


Figure 10. Basic Block diagram for Connected Component based text extraction.

3.2.1 Pre-Processing

This section corresponds to Step 1 of 3.2. The input image is pre-processed to facilitate easier detection of text regions. As proposed in [3], the image is converted to the YUV color space (luminance + chrominance), and only the luminance(Y) channel is used for further processing. The conversion is done using the MATLAB function 'rgb2ycbcr' which takes the input RGB image and converts it into the corresponding YUV image. The individual channels can be extracted from this new image. The Y channel refers to brightness or intensity of the image whereas the U and the V channels refer to the actual color information [12]. Since text present in an image has more contrast with its background, by using only the Y channel, the image can be converted to a grayscale

image with only the brightness / contrast information present. Figure 11(2, 3, 4) show the Y, U and V channels respectively for an input test image [1,2] in (1).

(1) Original Image [1,2]

(2) Y channel

(3) U channel

(4) V channel

Figure 11. YUV channels for test image (1)

3.2.2 Detection of edges

This section corresponds to Step 2 of 3.2. In this process, the connected-component based approach is used to make possible text regions stand out as compared to non-text regions. Every pixel in the edge image is assigned a weight with respect to its neighbors in each direction. As depicted in Figure 12, this weight value is the maximum value between the pixel and its neighbors in the left (L), upper (U) and upper-right (UR) directions [3]. The algorithm proposed in [3] uses these three neighbor values to detect edges in horizontal, vertical and diagonal directions. The resultant edge image obtained is sharpened in order

to increase contrast between the detected edges and its background, making it easier to extract text regions. Figure 13 below shows the sharpened edge image for the Y Channel gray image G from Figure 11, obtained by the algorithm proposed in [3].

The algorithm for computing the edge image E, as proposed in [3] is as follows:

1. Assign left, upper, upperRight to 0.
2. For all the pixels in the gray image G(x,y) do
    a. left = (G(x,y) – G(x-1,y))
    b. upper = (G(x,y) – G(x,y-1))
    c. upperRight = (G(x,y)-G(x+1,y-1))
    d. E(x,y) = max(left, upper, upperRight)
3. Sharpen the image E by convolving it with a sharpening filter.

W (x,y) = max(L,U,UR) x

	y	
UL	U	UR
L		R
BL	B	BR

Figure 12. Weight for pixel (x,y)

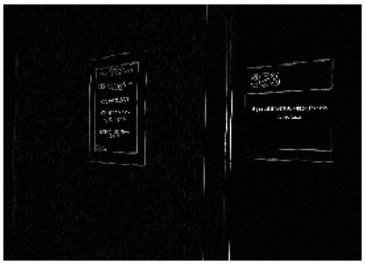
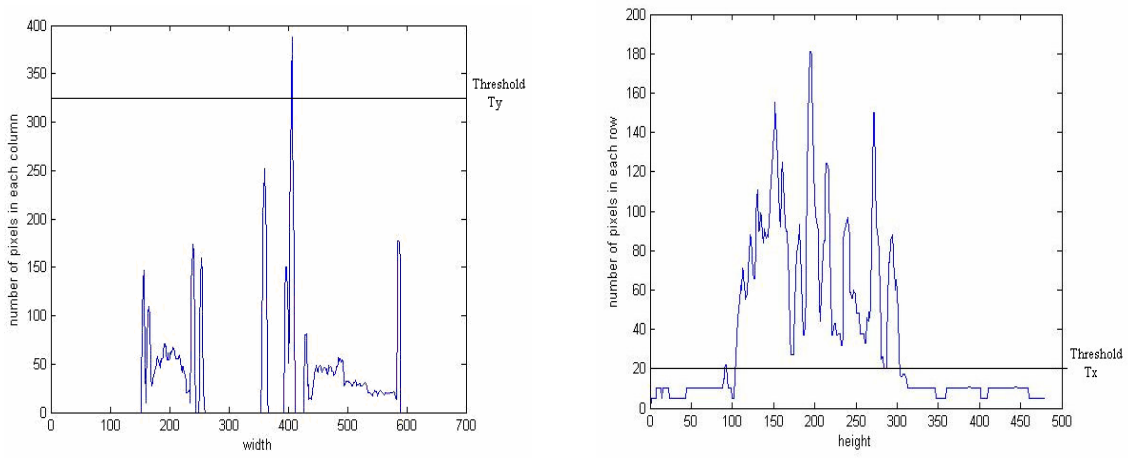


Figure 13. Sharpened Edge Image

3.2.3 Localization

This section corresponds to Step 3 of 3.2. In this step, the horizontal and vertical projection profiles for the candidate text regions are analyzed. The sharpened edge image is considered as the input intensity image for computing the projection profiles, with white candidate text regions against a black background. The vertical projection profile shows the sum of pixels present in each column of the intensity or the sharpened image. Similarly, the horizontal projection profile shows the sum of pixels present in each row of the intensity image. These projection profiles are essentially histograms where each bin is a count of the total number of pixels present in each row or column. The vertical and horizontal projection profiles for the sharpened edge image from Figure 13, are shown in Figure14 (a) and (b) respectively.



(a) **(b)**

Figure14. (a) Vertical Projection profile (b) Horizontal Projection profile for Sharpened image in Figure 13.

Candidate text regions are segmented based on adaptive threshold values, Ty and Tx, calculated for the vertical and horizontal projections respectively. Only regions that fall within the threshold limits are considered as candidates for text. The value of threshold Ty is selected to eliminate possible non text regions such as doors, window edges etc. that have a strong vertical orientation. Similarly, the value of threshold Tx is selected to eliminate regions which might be non text or long edges in the horizontal orientation.

$$Tx = \frac{\text{Mean(Horizontal projection profile)}}{20} \quad (3)$$

$$Ty = \text{Mean(Vertical projection profile)} + \frac{\text{Max (Vertical projection profile)}}{10} \quad (4)$$

3.2.4 Enhancement and Gap Filling

This section corresponds to Steps 4 to 6 of 3.2. The geometric ratio between the width and the height of the text characters is considered to eliminate possible non-text regions. This ratio value will be defined after experimenting on different kinds of images to get an average value. In this project, regions with minor to major axis ratio less than 10 are considered as candidate text regions for further processing. Next a gap image will be created which will be used as a reference to refine the localization of the detected text regions [3]. If a pixel in the binary edge image created is surrounded by black (background) pixels in the vertical, horizontal and diagonal directions, this pixel is also

substituted with the background value. This process is known as *gap filling*. An example of extracted text using this technique is shown in Figure 15.

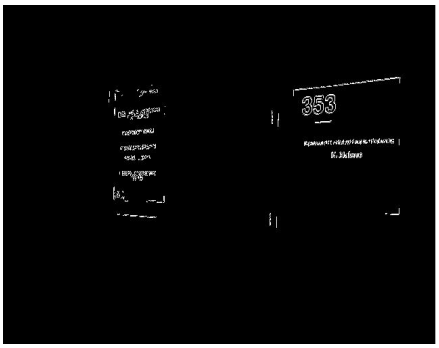


Figure 15. Result obtained by connected component based text detection algorithm for test image in Figure 11 (1)

4. Experiments / Results

The experimentation of the proposed algorithm was carried out on a data set consisting of different images such as indoor, outdoor, posters etc. These test images vary with respect to scale, lighting and orientation of text in the image. Currently the data set consists of 10 images provided by Xiaoqing Liu and Jagath Samarabandu, 2 CAPTCHA images [13], 2 Hindi character images [14], and variations of 2 indoor poster images as well as outdoor images, taken from a digital camera. The complete list of test images is shown in Table1. The significance of testing the algorithms on variations of scale, lighting and orientation is to determine the robustness of each technique with respect to variance in these conditions, and also to determine where each technique is successful and where it fails.

The performance of each technique has been evaluated based on its precision and recall rates obtained. As explained in the earlier sections, precision and recall rates are calculated as follows:

$$\text{Precision Rate} = \frac{\text{Correctly detected words}}{\text{Correctly detected words} + \text{False positives}} \times 100\% \quad (1)$$

$$\text{Recall Rate} = \frac{\text{Correctly detected words}}{\text{Correctly detected words} + \text{False Negatives}} \times 100\% \quad (2)$$

Precision rate takes into consideration the false positives, which are the non-text regions in the image and have been detected by the algorithm as text regions. Recall rate takes into consideration the false negatives, which are text words in the image, and have not been detected by the algorithm. Thus, precision and recall rates are useful as measures to determine the accuracy of each algorithm in locating correct text regions and eliminating non-text regions.

Scale Variance: The test images are varied with respect to the distance from the camera. This test is to evaluate the robustness of each algorithm with respect to size of text in an image. A total of 3 scale levels have been considered for each image type.

Lighting Variance: The test images are varied with respect to lighting conditions. This test is to evaluate the robustness of each algorithm to detect text with invariance to brightness in an image. The algorithms are run on images with 3 different lighting

conditions, for indoor as well as outdoor images. For the indoor images, white light, yellow light and no light conditions are used. For the outdoor images, day light, evening light and night conditions are used.

Orientation / Rotation Variance: The test images are varied with respect to the angle from the camera. Each image is rotated approximately 45 ° and 135 ° angles. This test evaluates the invariance of each algorithm with respect to rotation.

The precision and recall rates obtained by each algorithm, under different conditions of scale, lighting and rotation have been calculated and shown in the following sections. The results obtained from other images in the database have also been listed.

Type of image	**Scale**	**Lighting**	**Orientation / Rotation**	**Total number of images**
Digital Camera Indoor	3	3	2	8
Digital Camera Outdoor	3	3	2	8
CAPTCHA				2
Hindi Character				2
Authors [1,2]				10
Total				30

Table 1. Test Image Data Set

The precision and recall rates calculated for the connected component algorithm proposed in [7] takes into consideration each text line as one text region. The edge based algorithm proposed in [1,2] takes into consideration each character of text to calculate precision and recall rates. In order to have a common method to evaluate and compare the results from each algorithm, in this project each text word is considered in the calculation of precision and recall rate. False positives are the number of connected regions obtained by the algorithm, which are not text words. False negatives are the total number of text words in the test image minus the words which were not detected by the algorithm. The test image data set consists of 10 images provided by the authors of [1,2]. Most of the images used in this case are indoor images, with the exception of one outdoor image. It has been stated in [2] that the proposed edge based algorithm is robust with respect to illumination and orientation changes. The results obtained after implementation of the algorithm, as shown in the following section, indicate that it is less robust to lighting changes. The overall average precision and recall rates shown in [1,2] are over a varied data set of images. The average precision rate for the edge based algorithm as stated by the authors is 91.8% and the average recall rate is 96.6%. Only a small subset of the authors' database has been used for this project. The average precision rate obtained by this project for the edge based algorithm is 46.27% and the average recall rate obtained is 62.29%. Thus, the recall rate as stated is more than the precision rate obtained by the edge based algorithm. The overall average precision rate is 47.4% and average recall rate is 75.09%.

The test images used by [3] are mostly straightforward with no variations with respect to lighting or orientation considered. The proposed connected component based algorithm is stated to have an average precision rate of 88.7% and average recall rate of 83.9%. The results obtained after implementation of the proposed algorithm, as shown in the following section, gives an overall average precision rate of 50.10% and an average recall rate of 73.42%. Refer Table 9.

4.1 Scale Variance

Scale variance test is to determine the robustness of each algorithm to detect text regions for changes in scale or distance from the camera. The precision and recall rates obtained by both algorithms have been calculated for each of the three scaled test images as shown below.

4.1.1 Edge Based

Image Type	Image distance from camera (meters)	Precision Rate (%)	Recall Rate (%)
Indoor	(1) 0.4	63.07	85.41
	(2) 0.8	43.54	27.02
	(3) 1.2	64.51	93.02
Outdoor	(4) 1.5	20.68	75.00
	(5) 3.0	14.89	87.5
	(6) 4.5	7.27	50.00

Table 2. Results from edge based algorithm

4.1.2 Connected Component Based

Image Type	Image distance from camera (meters)	Precision Rate (%)	Recall Rate (%)
Indoor	(1) 0.4	68.25	89.58
	(2) 0.8	56.16	89.13
	(3) 1.2	58.33	81.39
Outdoor	(4) 1.5	23.07	75.00
	(5) 3.0	24.13	87.5
	(6) 4.5	16.27	87.5

Table 3. Results from connected component based algorithm

Tables 2 and 3 above show the results obtained by each algorithm for two different image types, varied with respect to distance from the camera, (1) being the closest and (3) being the farthest from the camera. In case of indoor images, the average precision rate obtained by the connected component based algorithm (60.91%) is higher than that obtained by the edge based algorithm (57.04%). Also, the recall rates obtained by the connected component algorithm (86.7%) are higher than those obtained by the edge based algorithm (68.48%). In case of outdoor images also, the average precision (21.15%) and recall (83.33%) rates obtained by the connected component based algorithm are higher than those obtained by the edge based algorithm (14.28%, 70.83%). Figure 16 shows three original indoor scaled images and respective results obtained from each algorithm.

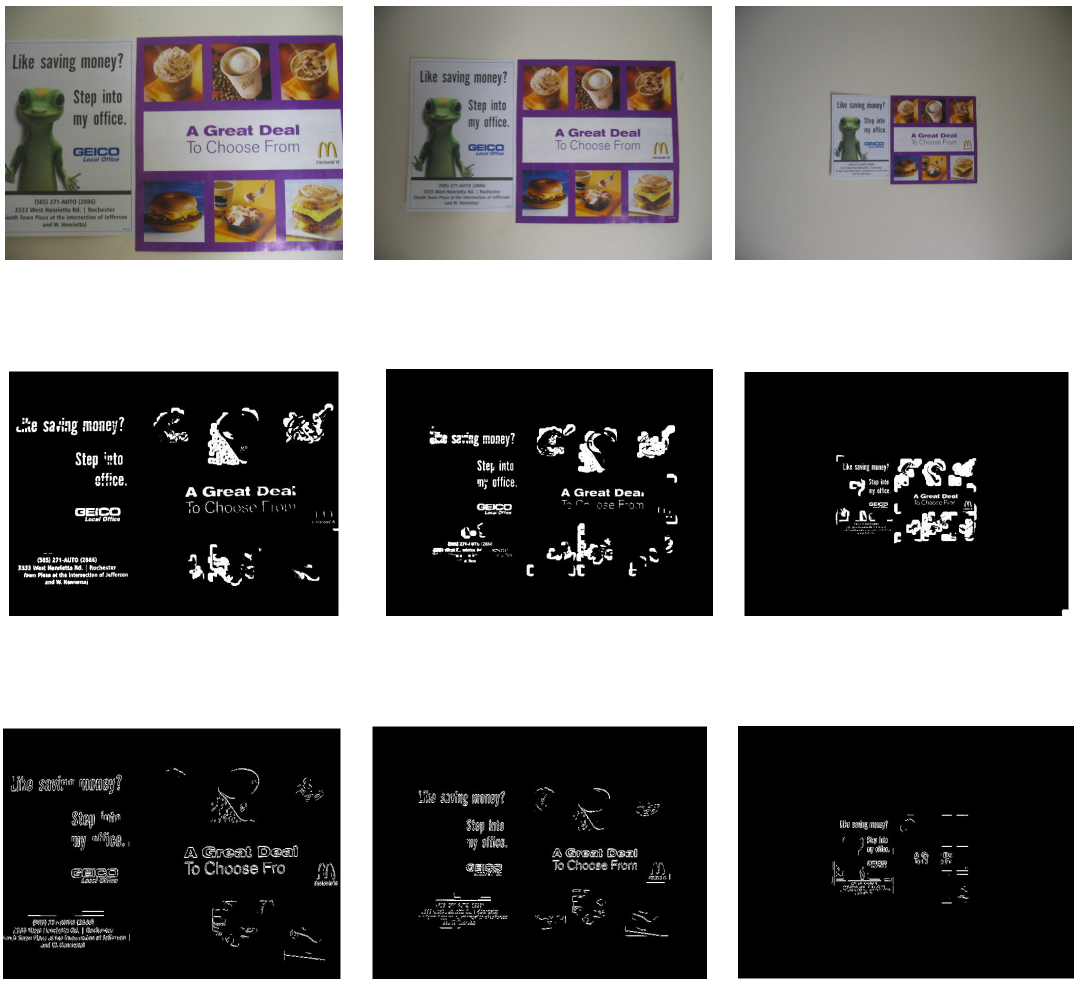
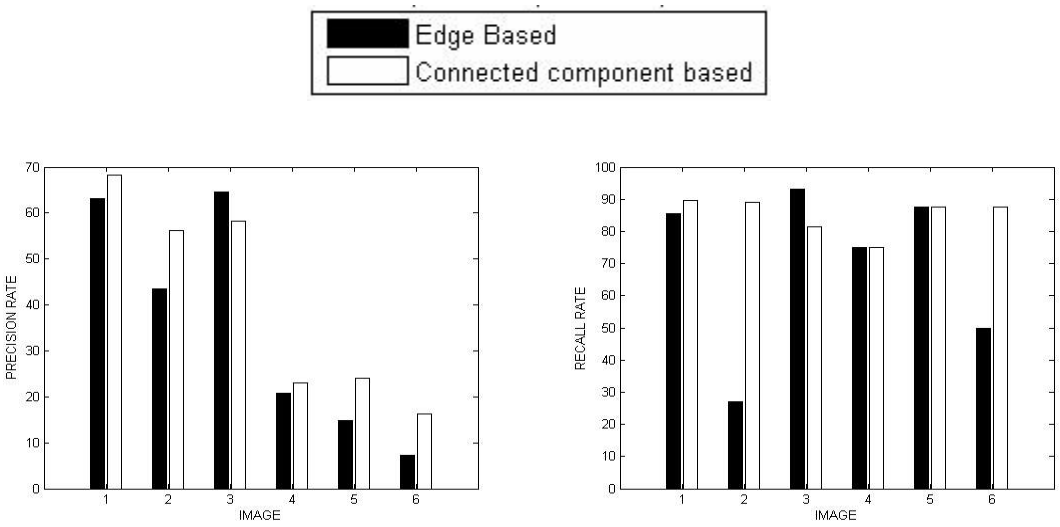


Figure 16. (Row 1) Original indoor images at three different scales (Row 2) Results from edge based algorithm (Row 3) Results from connected component based algorithm

The graph in Figure 17 below shows that the precision and recall rates obtained by the connected component based algorithm are higher than those obtained by the edge based algorithm. Thus, the connected component algorithm is more robust and invariant to scale changes as compared to the edge based algorithm for text region extraction, at least for the sample images tested.



(a) **(b)**

Figure 17. (a) Precision rates (b) recall rates for each scaled image in Tables 2 and 3

4.2 Lighting Variance

The lighting variance test is to determine the robustness or invariance of each algorithm to changes in lighting conditions. The precision and recall rates obtained from each algorithm have been shown below for three indoor and three outdoor images.

4.2.1 Edge Based

Image Type	Image at different lighting conditions	Precision Rate (%)	Recall Rate (%)
Indoor	(1) White light	66.66	83.33
	(2) Yellow light	21.33	33.33
	(3) No light	37.5	43.75
Outdoor	(4) Day light	34.78	100.00
	(5) Evening light	20.58	87.5
	(6) Night light	66.66	100.00

Table 4. Results from edge based algorithm

4.2.2 Connected Component Based

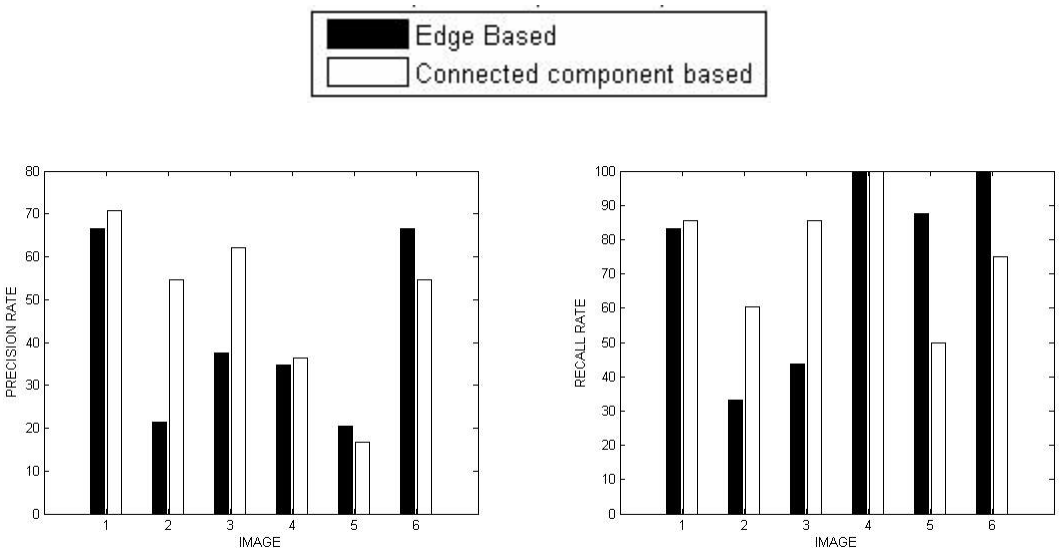
Image Type	Image at different lighting conditions	Precision Rate (%)	Recall Rate (%)
Indoor	(1) White light	70.68	85.41
	(2) Yellow light	54.71	60.41
	(3) No light	62.12	85.41
Outdoor	(4) Day light	36.36	100.00
	(5) Evening light	16.66	50.00
	(6) Night light	54.54	75.00

Table 5. Results from connected component based algorithm

Tables 4 and 5 above show the precision and recall rates obtained by each algorithm when tested for three different lighting conditions. For indoor lighting the following conditions were considered: (1) White light (2) Yellow light (3) No light. For outdoor images the following conditions were considered: (1) Day light (2) Evening light (3) Night. The results obtained show that the average precision rate in case of indoor images from the connected component based algorithm (62.50%) is higher than those from the edge based algorithm (41.83%). The average recall rate from the connected component algorithm (77.07%) is also higher than from the edge based algorithm (53.47%). In case of outdoor images, the average precision (40.67%) and recall (95.83%) rates obtained from the edge based algorithm are higher than those obtained by the connected component algorithm (35.85%, 75%). Thus, the connected component algorithm is more robust in indoor lighting conditions, and the edge based algorithm is more robust in

outdoor lighting conditions. Figure 18 below shows the results obtained by each algorithm under three different lighting conditions for an outdoor image.

Figure 18. (Row 1) Original images at three different lighting conditions (Row 2) Results from edge based algorithm (Row 3) Results from connected component based algorithm



(a) **(b)**

Figure 19. (a)Precision rates (b) Recall rates for lighting variance

The graph in Figure 19 above shows that the overall average precision and recall rates obtained by the connected component based algorithms are slightly higher than those obtained by the edge based algorithm. Thus, the connected component algorithm is a little more robust to lighting variance as compared to the edge based algorithm.

4.3 Orientation / Rotation Variance

This test is to evaluate the robustness of each algorithm for orientation or rotation variance of text in images. Three indoor and three outdoor images have been considered with 0 º, 45 º and 135 º rotation angles. The precision and recall rates obtained from each algorithm have been shown below.

4.3.1 Edge Based

Image Type	Image at different orientations (degrees)	Precision Rate (%)	Recall Rate (%)
Indoor	(1) 0	43.54	27.02
	(2) 45	59.92	92.5
	(3) 135	55.26	91.30
Outdoor	(4) 0	34.78	100.00
	(5) 45	25.00	100.00
	(6) 135	24.24	100.00

Table 6. Results from edge based algorithm

4.3.2 Connected Component Based

Image Type	Image at different orientations (degrees)	Precision Rate (%)	Recall Rate (%)
Indoor	(1) 0	56.16	89.13
	(2) 45	75.55	85.00
	(3) 135	51.92	58.69
Outdoor	(4) 0	36.36	100.00
	(5) 45	44.44	100.00
	(6) 135	16.00	100.00

Table 7. Results from connected component based algorithm

Tables 6 and 7 above shows the results obtained from each algorithm when tested for orientation or rotation invariance. In case of indoor images, the average precision rate obtained by the connected component algorithm (61.21%) is slightly higher than the average precision rate obtained by the edge based algorithm (52.90%). Also, the average recall rate obtained by the edge based algorithm (70.27%) is lesser than that obtained by the connected component based algorithm (77.60%). In case of outdoor images, the average precision rate obtained by the connected component based algorithm (32.26%) is higher than the average precision rate from the edge based algorithm (28%). The recall rates from each algorithm are 100%. Figure 20 below shows the results obtained from each algorithm.

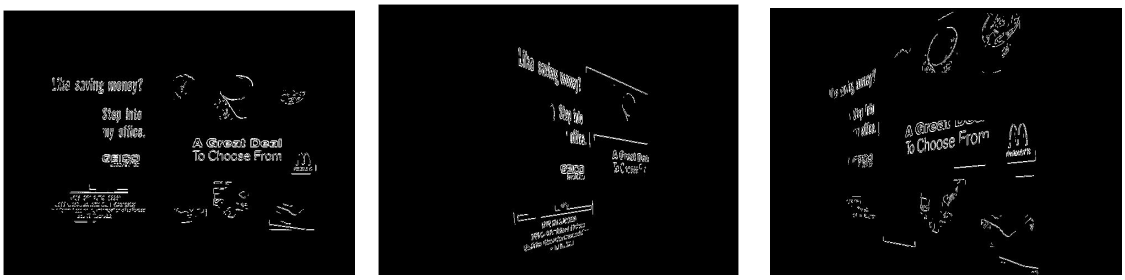
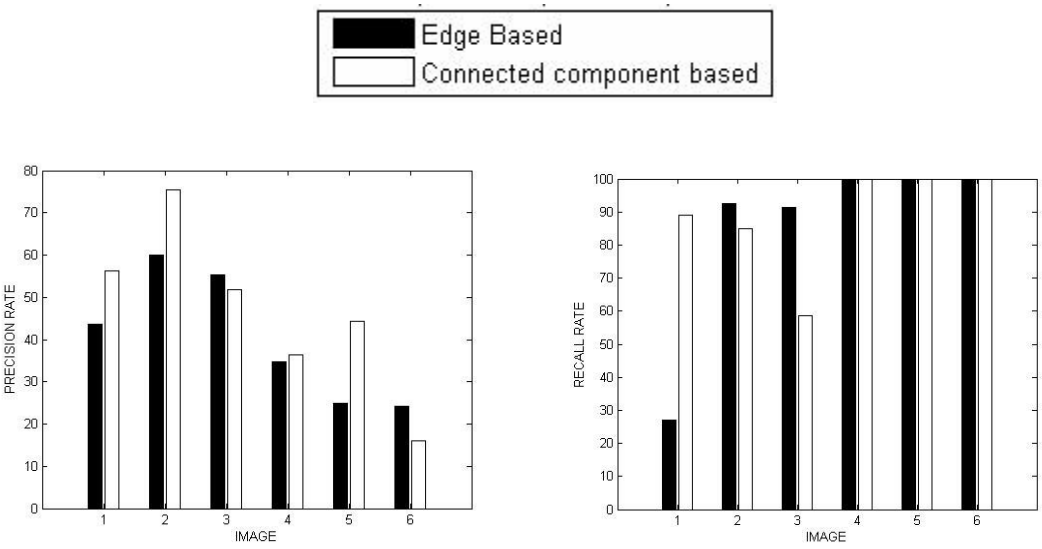


Figure 20. (Row 1) Original Images at three different rotation angles (Row 2) Results from edge based algorithm (Row 3) Results from connected component based algorithm



(a) **(b)**

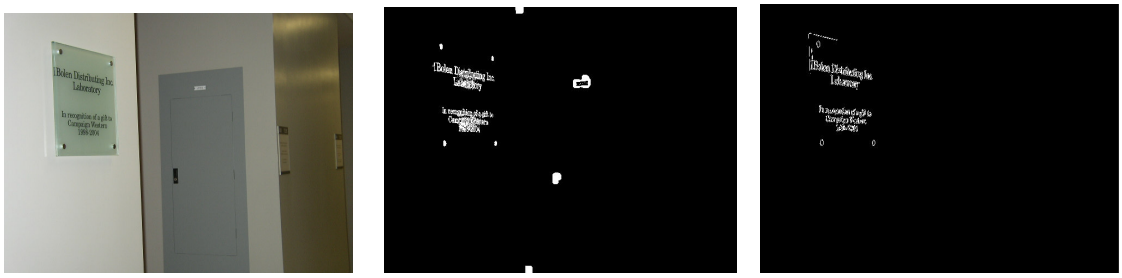
Figure 21. (a) Precision rates (b) Recall rates for orientation/rotation variance

As depicted by the bar graph in Figure 21 above, the average precision rates obtained by the connected component based algorithm are higher than the edge based. Also, the average recall rates obtained by the connected component based are higher than or equal to the edge based algorithm.

4.4 Results for other images

Image	Edge Based		Connected Component Based	
	Precision Rate (%)	Recall Rate (%)	Precision Rate (%)	Recall Rate (%)
16	14.28	27.27	18.51	45.45
32	20.83	26.31	20.00	10.52
35	35.71	100.00	60.00	60.00
113	62.22	100.00	75.75	89.28
lab1	71.42	51.72	73.68	48.27
lab2	43.24	55.17	63.63	48.27
lab3	90.90	100.00	74.35	72.50
lab16	16.66	33.33	31.81	58.33
lab24	46.15	50.00	46.66	29.16
sign1_1	61.29	79.16	33.80	100.00
Captcha1(34)	100.00	100.00	100.00	100.00
Captcha2(gimpy)	92.30	92.30	90.00	69.23
Hindi1	93.93	93.93	83.87	78.78
Hindi2	30.76	100.00	34.78	100.00

Table 8. Results for other images from edge based and connected component based algorithms (All images are shown in the appendix)

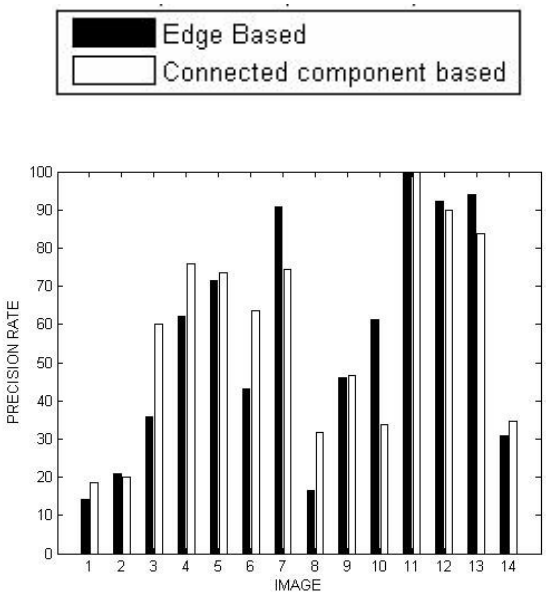


(a) **(b)** **(c)**

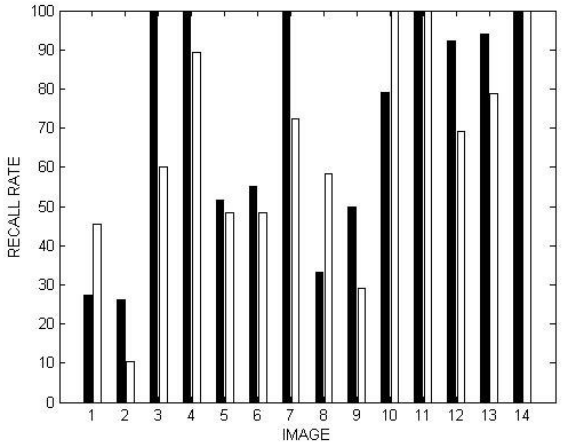
Figure 22. (a) Original image (lab1) from Xiaoqing Liu and Jagath Samarabandu (b) Result from edge algorithm (c) Result from connected component algorithm

(a) **(b)** **(c)**

Figure 23. (a) Original image (Captcha1(34)) (b) Result from edge based algorithm (c) Result from connected component algorithm



(a)



(b)

Figure 24. (a) Precision rates (b) Recall rates for other images in the test database

The results for precision and recall rates obtained by each algorithm when tested on the remaining images in the database have been listed in Table 8. The graph in Figure 24 above shows that the average precision rate obtained by the connected component (54.63%) and the edge based algorithm (55.69%) are very close to each other. The average recall rates obtained by the edge based algorithm (72.08%) are higher than those obtained by the connected component algorithm (64.98%).

	Precision Rate (%)	**Recall Rate (%)**
Edge algorithm	**47.4**	**75.09**
Connected component algorithm	**50.10**	**73.42**

Table 9. Overall precision and recall rates

5. Conclusion and Recommendations

The results obtained by each algorithm on a varied set of images were compared with respect to precision and recall rates. In terms of scale variance, the connected component algorithm is more robust as compared to the edge based algorithm for text region extraction. In terms of lighting variance also, the connected component based algorithm is more robust than the edge based algorithm. In terms of rotation or orientation variance, the precision rate obtained by the connected component based algorithm is higher than the edge based, and the recall rate obtained by the edge based is higher than the connected component based The average precision rates obtained by each algorithm for the remaining test images are similar, whereas the average recall rate obtained by the connected component algorithm is a little lower than the edge based algorithm. Thus, the results from the experiments indicate that in most of the cases, the connected component based algorithm is more robust and invariant to scale, lighting and orientation as compared to the edge based algorithm for text region extraction. For the edge based algorithm, the overall precision rate is 47.4% and recall rate is 75.09%. For the connected component based algorithm, the overall precision rate is 50.10% and recall rate is 73.42%. Refer Table 9.

For future work the following recommendations can be taken into consideration:

1. Combining the edge and connected component based algorithms: Each of the algorithms is by itself quite robust in extracting text regions from natural images. A combination of these techniques can produce more efficient outputs. The results of

this project show that the connected component based algorithm is more robust to scale and lighting conditions as compared to the edge based algorithm. Also, the results obtained by each algorithm for rotation variance are similar. Using a combination of the two approaches, a far more robust algorithm can be achieved, which would be invariant to scale, lighting as well as orientation changes.

2. Morphological cleaning of images: The approach used by the edge based as well as the connected component based algorithm does not take into consideration the removal or noise or unwanted clutter from the test images before or after the computations. A morphological cleaning operation would be helpful in reducing the number of false positives obtained. Thus, cleaning of the image could result in a higher precision rate.

3. Testing on different kind of images: The goal of this project was to compare the two algorithms for scale, lighting and orientation variance in natural images. In order to more thoroughly evaluate these techniques, the algorithms can be tested on different kind of images and video frames such as movie clips, animated scenes, comic book images, as well as document images.

4. Testing for hidden text region extraction in a cluttered scene: An interesting test would be to find text regions which are hidden behind other objects or water marked within an image. In order to achieve this, various other approaches mentioned in the earlier sections can be explored.

References

[1] Xiaoqing Liu and Jagath Samarabandu, *An Edge-based text region extraction algorithm for Indoor mobile robot navigation*, Proceedings of the IEEE, July 2005.

[2] Xiaoqing Liu and Jagath Samarabandu, *Multiscale edge-based Text extraction from Complex images*, IEEE, 2006.

[3] Julinda Gllavata, Ralph Ewerth and Bernd Freisleben, *A Robust algorithm for Text detection in images*, Proceedings of the 3rd international symposium on Image and Signal Processing and Analysis, 2003.

[4] Keechul Jung, Kwang In Kim and Anil K. Jain, *Text information extraction in images and video: a survey*, The journal of the Pattern Recognition society, 2004.

[5] Kongqiao Wang and Jari A. Kangas, *Character location in scene images from digital camera*, The journal of the Pattern Recognition society, March 2003.

[6] K.C. Kim, H.R. Byun, Y.J. Song, Y.W. Choi, S.Y. Chi, K.K. Kim and Y.K Chung, *Scene Text Extraction in Natural Scene Images using Hierarchical Feature Combining and verification*, Proceedings of the 17th International Conference on Pattern Recognition (ICPR '04), IEEE.

[7] Victor Wu, Raghavan Manmatha, and Edward M. Riseman, *TextFinder: An Automatic System to Detect and Recognize Text in Images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 11, November 1999.

[8] Xiaoqing Liu and Jagath Samarabandu, *A Simple and Fast Text Localization Algorithm for Indoor Mobile Robot Navigation*, Proceedings of SPIE-IS&T Electronic Imaging, SPIE Vol. 5672, 2005.

[9] Qixiang Ye, Qingming Huang, Wen Gao and Debin Zhao, *Fast and Robust text detection in images and video frames*, Image and Vision Computing 23, 2005.

[10] Rainer Lienhart and Axel Wernicke, *Localizing and Segmenting Text in Images and Videos*, IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, No.4, April 2002.

[11] Qixiang Ye, Wen Gao, Weiqiang Wang and Wei Zeng, *A Robust Text Detection Algorithm in Images and Video Frames*, IEEE, 2003.

[12] http://softpixel.com/~cwright/programming/colorspace/yuv/

[13] http://www.captcha.net

[14] http://images.google.com

Appendix

List of test images

4035

4029

4033

3998

4002

3999

4053

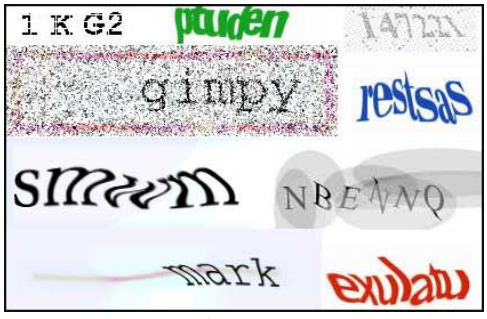
4037

16

32

35

113



Captcha

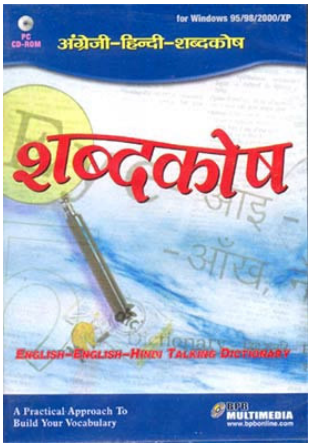
sign1_1

lab2

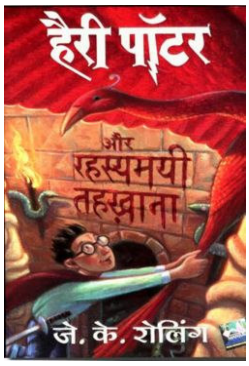
lab24

lab3

lab16

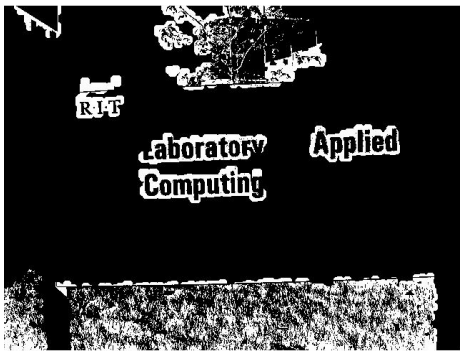


Hindi1

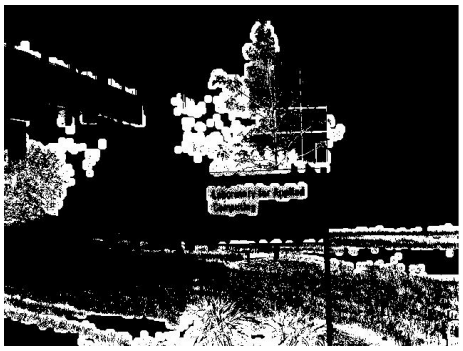


Hindi2

Results from edge based algorithm

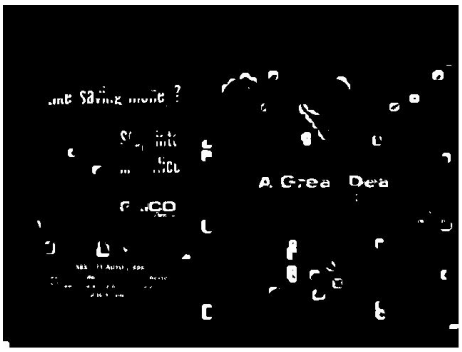


E_4035



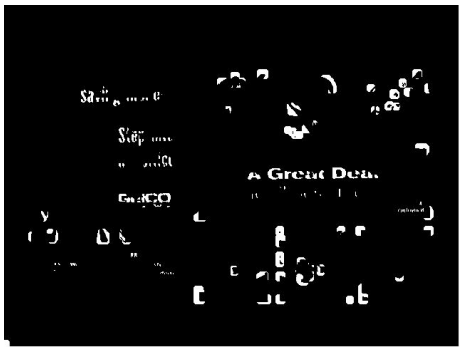
E_4029

E_4033

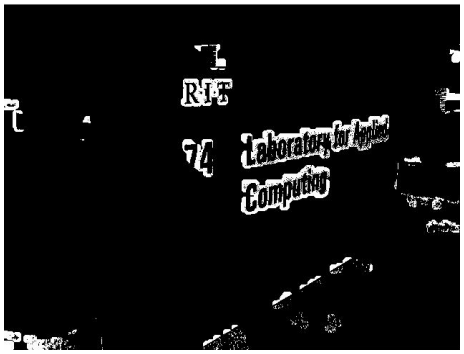


E_3998

E_4002

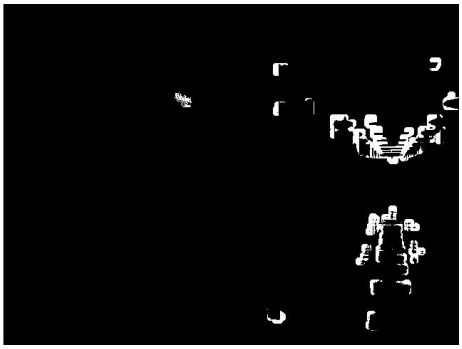


E_3999



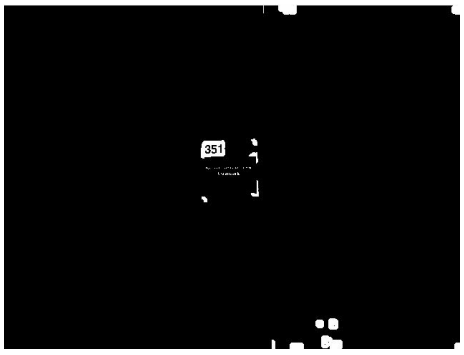
E_4053

E_4037



E_16

E_32

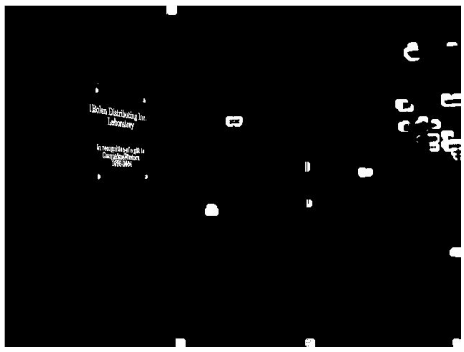


E_35

E_113

E_captcha

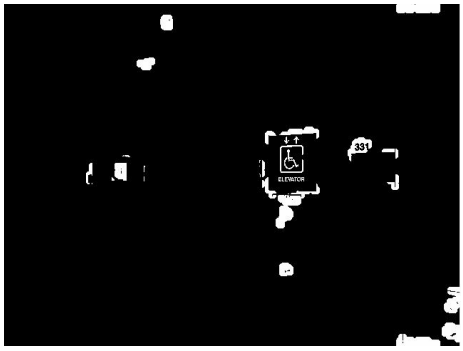
E_sign1_1



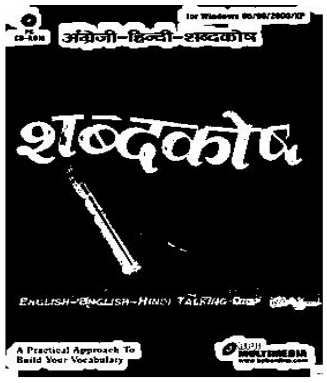
E_lab2

E_lab24

E_lab3



E_lab16

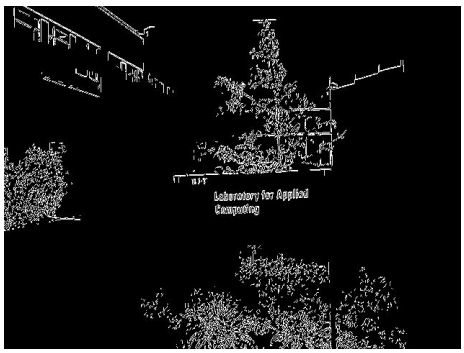


E_hindi1

E_hindi2

Results from connected component based algorithm

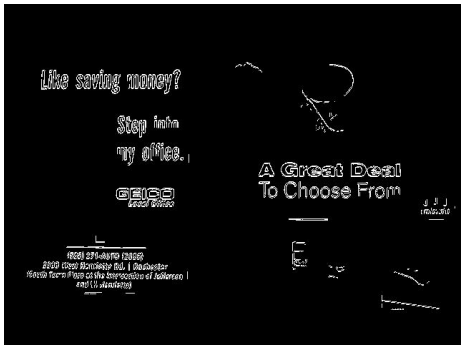
C_4035



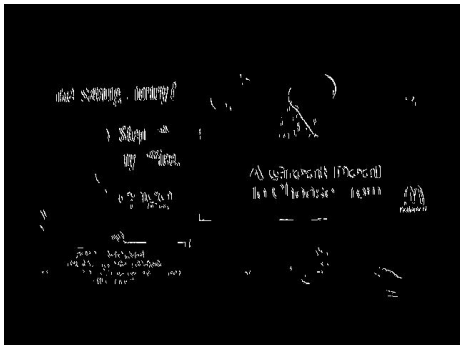
C_4029

C_4033

C_3998

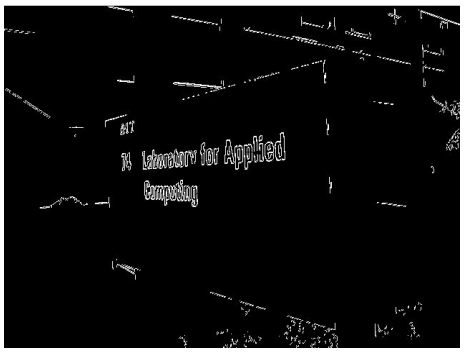


C_4002



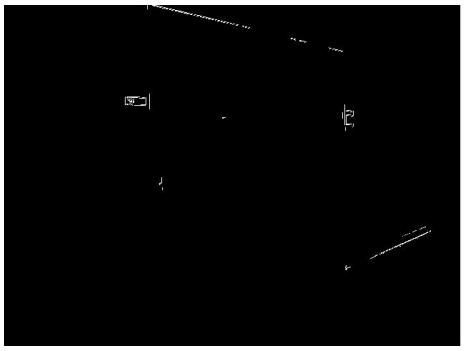
C_3999

C_4053

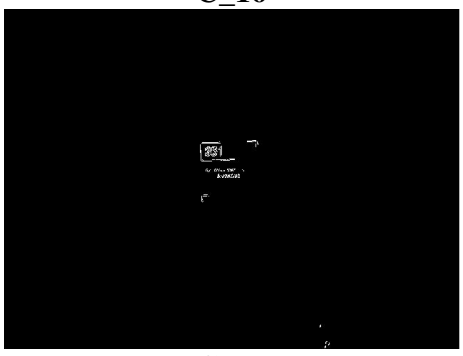


C_4037

C_16

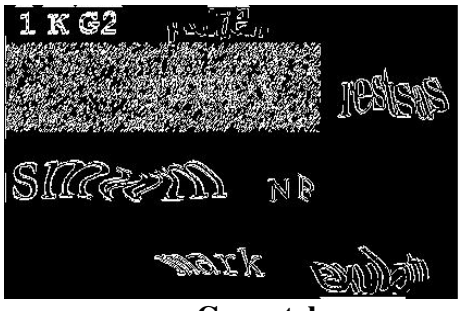


C_32



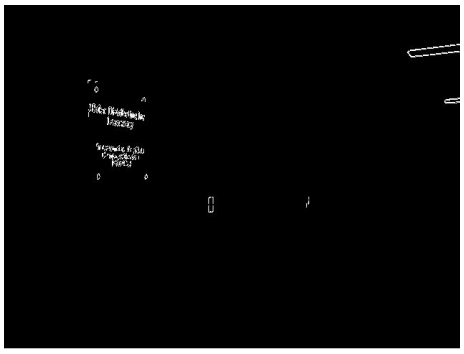
C_35

C_113

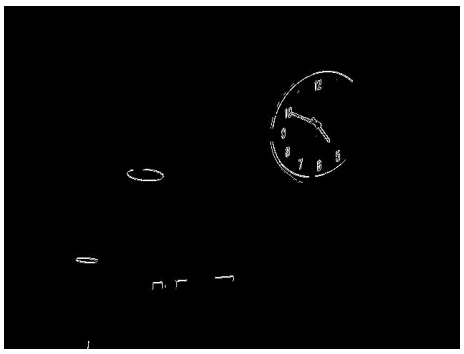


C_captcha

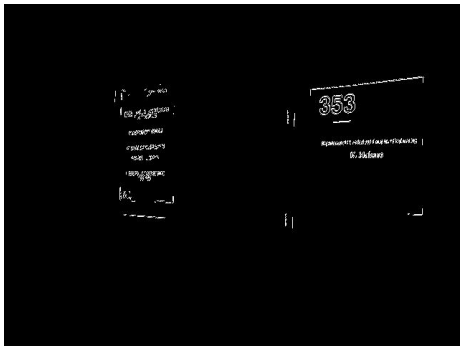
C_sign1_1



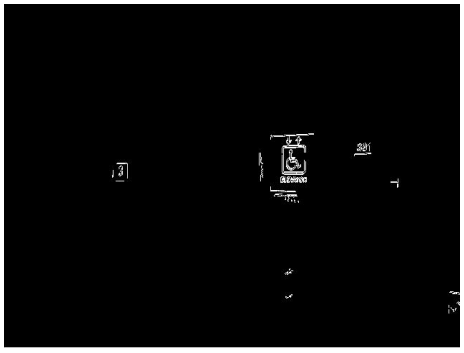
C_lab2



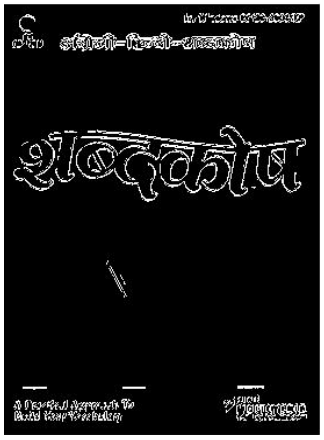
C_lab24



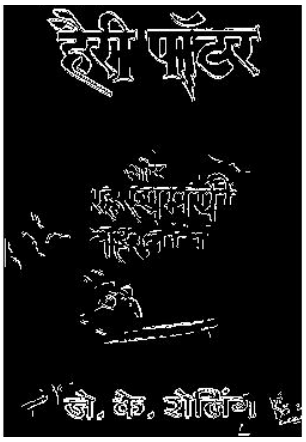
C_lab3



C_lab16



C_hindi1



C_hindi2

Matlab code for edge based algorithm

```
% Edge.m
%
% Edge Based Text region extraction algorithm
%
% Author: Sneha Sharma
%

% Read the input image
I = imread('IMG.JPG');
Ibin = im2bw(I);

% The direction filters

kernel0 = [-1 -1 -1
   2 2 2
   -1 -1 -1];  %0 degree

kernel45 = [-1 -1 2
   -1 2 -1
   2 -1 -1];   %45 degree

kernel90 = [-1 2 -1
   -1 2 -1
   -1 2 -1];   %90 degree

kernel135 = [2 -1 -1
   -1 2 -1
   -1 -1 2];   %135 degree

Kernels{1} = kernel0;
Kernels{2} = kernel45;
Kernels{3} = kernel90;
Kernels{4} = kernel135;

% Creating Gaussian Pyramid

h = fspecial('gaussian');   %Gaussian kernel default hsize 3x3

im = I;
Pyramid{1} = im;
for i = 2:4
   im = imfilter(im,h,'conv'); %convolve with gaussian filter
   im = imresize(im,0.5);      %down-sample by 1/2
   Pyramid{i} = im;
```

```
   %figure,imshow(Pyramid{i});
end

% Convolving images at each level in the Pyramid with each
% direction filter

for m = 1:4
   for n = 1:4
      Conv{m,n} = imfilter(Pyramid{m},Kernels{n},'conv');
   end
end

% Resize images to original image size

for m = 1:4
   for n = 1:4
      Conv2{m,n} = imresize(Conv{m,n},[size(I,1) size(I,2)]);
   end
end

% Total of all directional filter responses

for m = 1:4

   total{m} = im2bw(Conv2{1,m}+Conv2{2,m}+Conv2{3,m}+Conv2{4,m});
end

Total = imadd((total{1,1}+total{1,3}),(total{1,2}+total{1,4}));
%figure,imshow(Total),title('Total of directions');

% Otsu threshold
level = graythresh(double(total{1,3}));
EdgeStrong = im2bw(total{1,3},level);
%figure,imshow(EdgeStrong),title('Strong');

%dilation with SE 1x3
SE = strel('line',3,0);
IDilated = imdilate(EdgeStrong,SE);
%figure,imshow(IDilated),title('Dilated');

%Closing with vetical SE
m = round(size(EdgeStrong,1)/25);
SE2 = strel('line',m,90);
IClosed = imclose(IDilated,SE2);
%figure,imshow(IClosed),title('Closed');
% Weak edges
```

```
EdgeWeak = IClosed-IDilated;
%figure,imshow(EdgeWeak),title('Weak');

%Combining strong and weak edges
Edge90 = EdgeStrong + EdgeWeak;
%figure,imshow(Edge90),title('Edge90');

%Thinning operation
Thinned = bwmorph(Edge90,'thin',Inf);
%figure,imshow(Thinned),title('Thinned');

% Eliminate long edges
[L,N] = bwlabel(Thinned,4);
St = regionprops(L,'all');
Short90 = double(Thinned);

for i=1:length(St)
   if St(i).MajorAxisLength > (size(I,1)/5)

      c = St(i).PixelList(:,1);
      r = St(i).PixelList(:,2);
      Short90(r,c)=0;
   end
end
%figure,imshow(Short90),title('Short edges');

SED = strel('line',5,90);
candidate = imdilate(Short90,SED);
%figure,imshow(candidate),title('Candidate');

Refined = immultiply(candidate,Total);
%figure,imshow(Refined),title('refined');
ref = imdilate(Refined,strel('square',4));

%Feature Map
bic0 = im2bw(total{1,1});
bic90 = im2bw(total{1,3});
bic45 = im2bw(total{1,2});
bic135 = im2bw(total{1,4});

T1 = (bic0 & bic90);
T2 = (bic45 & bic135);

T = T1 + T2;
%figure,imshow(T),title('AND result');
FeatureMap = (ref&T);
```

```
%figure,imshow(FeatureMap),title('Feature Map');

BigSE2 = strel('disk',6);
FMDilated = imdilate(FeatureMap,BigSE2);
%figure,imshow(FMDilated),title('Dilated Feature Map');

% Heuristic Filtering
% Remove those regions which have Area < MaxArea/20
% Remove those regions which have Width/Height < 0.1

[Lab,Num] = bwlabel(FMDilated,4);
Regions = regionprops(Lab,'all');
MaxArea = 0;

for r=1:length(Regions)

   Area = Regions(r).Area;
   if(MaxArea < Area)
      MaxArea = Area;
   end
end
i=1;
for r=1:length(Regions)

   A = Regions(r).Area;
   if(A < MaxArea/20)

      FMDilated = bwareaopen(FMDilated,A);
   end

end

NewImage = double(FMDilated);

for i=1:length(Regions)
   if (Regions(i).MajorAxisLength / Regions(i).MinorAxisLength)>6

      c = Regions(i).PixelList(:,1);
      r = Regions(i).PixelList(:,2);
      NewImage(r,c)=0;
   end
end
%figure,imshow(NewImage);


% Final result
```

```
Final = immultiply(~(Ibin),im2bw(NewImage));
figure,imshow(Final),title('Result');
```

Matlab code for connected component algorithm

```
% CC.m
%
% Connected Component Based Text region extraction algorithm
%
% Author: Sneha Sharma
%

I = imread('IMG.JPG');

% Convert to YUV color space
yuv = rgb2ycbcr(I);

YChannel = yuv(:,:,1); % Y Channel
%figure,imshow(YChannel);

% Generate Edge Image from Gray Image

X=size(YChannel,1);
Y=size(YChannel,2);
x=0;
y=0;
left=0;
upper=0;
rightUpper=0;

for x=2:size(YChannel,1)-1
   for y=2:size(YChannel,2)-1

      if(( 0<x<X )&(0<y<Y))
         left =  imabsdiff(YChannel(x,y),YChannel(x-1,y));

         upper = imabsdiff(YChannel(x,y),YChannel(x,y-1));

         rightUpper = imabsdiff(YChannel(x,y),YChannel(x+1,y-1));

         YEdge(x,y) = max(max(left,upper),rightUpper);

      else
```

```
            YEdge(x,y) = 0;
        end

    end
end

%figure,imshow(YEdge);

% Increase contrast by sharpening
H = fspecial('unsharp');
sharpEdge = imfilter(YEdge,H,'replicate');
%figure,imshow(sharpEdge),title('Sharpened Edge Image');

gt = graythresh(sharpEdge);
b = im2bw(sharpEdge,gt);
b = bwareaopen(b,4);
b1 = imdilate(b,strel('rectangle',[2 5]));
%figure,imshow(b1);

% Calculate Horizontal and vertical projection profiles

S1 = sum(b1,1);  % vertical y
S2 = sum(b1,2);  % horizontal x

axis([0 length(S1) 0 max(S1)])
plot(S1),xlabel('width'),
ylabel('number of pixels in each column');   %vertical projection
%stem stem3 bar

axis([0 max(S2) 0 length(S2)])
plot(S2),xlabel('height'),
ylabel('number of pixels in each row');       %horizontal projection


Ty = mean(S1) + max(S1)/10;  %Vertical threshold

% Supress all pixels with value > Ty
for i=1:length(S1)
    if S1(i) > Ty
        S1(i)=0;
    end
end


VEdge = zeros(size(b1));
for y=1:size(VEdge,1)
```

```
   for x=1:length(S1)
      if( S1(x) == 0 )
         VEdge(y,x) = 0;
      else
         VEdge(y,x) = b1(y,x);
      end
   end
end
%figure,imshow(VEdge),title('Vertical Projection pixels');

Tx = mean(S2)/20;   %horizontal thresh

% Supress all pixels with value < Tx
for j=1:length(S2)
   if S2(j) < Tx
      S2(j)=0;
   end
end


HEdge = zeros(size(b1));
if (size(b1,1)<size(b1,2))
   for x=1:size(HEdge,1)
      for y=1:length(S2)
         if( S2(y) == 0 )
            HEdge(x,y) = 0;
         else
            HEdge(x,y) = b1(x,y);
         end
      end
   end
else
   for y=1:length(S2)
      for x=1:size(HEdge,2)
         if( S2(y) == 0 )
            HEdge(y,x) = 0;
         else
            HEdge(y,x) = b1(y,x);
         end
      end
   end
end

%figure,imshow(HEdge),title('Horizontal Projection pixels');

TotalEdge = imadd(HEdge,VEdge);
```

```
%figure,imshow(TotalEdge);

medFilt = medfilt2(TotalEdge,[4 4]);
%figure,imshow(medFilt),title('Noise Removed');


Final = immultiply(b,medFilt);
%figure,imshow(Final);

HSE = strel('line',10,90);
VSE = strel('line',10,0);
Final1 = imopen(Final,HSE);
Final2 = imopen(Final,VSE);
newFinal = Final-(Final1+Final2);
newFin = bwmorph(newFinal,'majority');
newFin = imdilate(newFin,strel('disk',6));
%figure,imshow(newFin);

% Segment out non-text regions using major to minor axis ratio

[Lab,N] = bwlabel(newFin,4);
Regions = regionprops(Lab,'all');
MaxArea = 0;

for r=1:length(Regions)
   Area = Regions(r).Area;
   if(MaxArea < Area)
      MaxArea = Area;
   end
end

for r=1:length(Regions)
   A = Regions(r).Area;
   if(A < MaxArea/20)
      newFin = bwareaopen(newFin,A);
   end
end
%figure,imshow(newFin);

[newLab,newN] = bwlabel(newFin,4);
newRegions = regionprops(newLab,'all');
J = double(newFin);
for r=1:length(newRegions)
   major = newRegions(r).MajorAxisLength;
   minor = newRegions(r).MinorAxisLength;
```

```
    R = major/minor;
    if(R>10)

        PListx = newRegions(r).PixelList(:,1);
        PListy = newRegions(r).PixelList(:,2);

        J(PListy,PListx)=0;
    end
end
%figure,imshow(J);


RR = imerode(J,strel('line',3,90));
RR = imdilate(RR,strel('disk',5));

FinalRes = immultiply(b,RR);
figure,imshow(FinalRes),title('Result');
```